Journal of Theoretical and Applied Information Technology
20th June 2014. Vol. 64 No.2
© 2005 - 2014 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# AN EFFICIENT DATA HIDING USING HEXADECIMAL SUDOKU FOR STEGANOGRAPHY

[1]PREMAMAYUDU BULLA, [2]RAGHUNATH AKKINENI, [3]SUBBA RAO PERAM, [4]NAGABHUSHANAM DASARI
Vignan's Foundation for Science, Technology and Research University, Vadlamudi. Andhra Pradesh, India.
E-mail: [1]premamayudu@gmail.com, [2]akkineniraghunath@gmail.com , [3]subbarao_peram@yahoo.com, [4]bhushanamdot@gmail.com

## ABSTRACT

This paper proposed an improved data hiding steganographic scheme using Sudoku puzzle, which can be utilize to identify least distortion pixel value of selected pixel from the digital image with respect to secret digit. The scheme calculates three components for the selected pixel from the Sudoku puzzle based on the secret digit. The proposed scheme does not affect the visual quality of the stego-image. This work is simulated in the MAT LAB and presented results. The performance of scheme is compared with the existing methods with respect to image distortion. The quality of the steg-image is also verified using image assessment parameters PSNR and MSE.

**Keywords:** *Image Steganography, Data Hiding, Steganalysis, Sudoku Solution, Stego-image.*

## 1. INTRODUCTION

Steganography or “hidden writing” is a technique which hides the binary data in digital media with the addition of very few noticeable changes [14]. Its goal is to achieve secrete communication through digital media e.g. image, video and audio files between two parties those are interested in hiding. To overcome the point of attack here is always concealing the very existence of the hidden data into the carrier. The hiding is designed to achieve efficient tradeoffs among the three conflicting goals of maximizing rate of information hiding, minimizing distortion between the host and composite signal, and maximizing the robustness of hiding. For decades people strove to develop innovative methods using steganographic approaches for secrete communication [19]. Steganography have a long and exciting history that goes back to antique. It can be detected back to 440 BC in ancient Greece. They were used a technique for writing a secrete message on a wooden panel cover it by a wax, and then hide a secret message on the wax. In another situation where steganography was used by spies, prisoners, and solders during World War II because mail was seriously inspected. So many instances were happened, in case of postal censors crossed out anything that looked like secret/sensitive information and prosecuted individuals for suspicious mail [18]. To prevent the secrete message from being delivered, censors even randomly deleted innocent-looking sentences or entire paragraph.

Recently the interest is increasing in the digital Steganography on carrier multimedia image. Image trafficking for commercial applications is increasing day by day in internet [13][15]. For the exponential growth of potential computer user, secrete communication and image authentication using Steganographic approaches are the demanded area of research. In multimedia applications, image authentication techniques got great attention due to its importance [10]. Many applications are using images for authentication and transmitting over unsecure channels such as the internet. Therefore, medical, military and quality control images must be secured from manipulations.

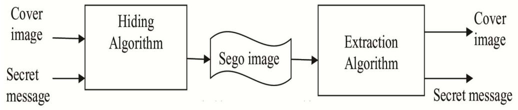


*Figure 1: A Typical Image Steganographic Process*

To shield the authenticity of multimedia images, several schemes have been proposed [16]. These methods include cryptography, watermarking and digital signatures that are based on the type of image. Many image authentication and identification techniques are implemented through Data hiding. Figure 1 shows the typical Steganographic scheme [12].

## 2. EXISTING SYSTEMS

AmitavaNag et. al. [1] have proposed Image seganography based on LSB using x-Box mapping. In this scheme, they used four x-boxes with sixteen different values. Each Box is operated with four bits. Each value compressed with four bits is mapped to the four LSBs of the cover image. They proposed to embed secret image into a gray-level cover image. The mapping rules used with x-Boxes are secret known by only the users of the steganography.

A.Cheddad et. al. [2] have surveyed on digital image steganography. The methods using DCT, DWT and Adaptive Steganography are not too prone to attacks, when the hidden message is too small. When payload is maximized in the cover image, it is very easy to identify the distortion.

Arupkumar pal, Tarok Pramanik [20], were proposed a method for improving the hiding capacity on gray scale cover image using LSB replacement. They exploit the presence of edges in the cover image to hide maximum amount of secret message digits. In this scheme, the cover image divided into edge region and non-edge region. After classification, x LSBs replaced in the pixels belongs to edge region and y LSBs replaced in the pixels of non-edge region where x>y.

Mielikainen [4] Proposed data hiding scheme using LSB Matching revisited method. They applied the embedding scheme using the grayscale cover image. Given an M x N sized grayscale cover image I = $p_{1,}p_2$ …….$p_{MxN}$ and a secret message bit stream S = $s_1$, $s_2$,….,$b_k$ where M and N are the cover image height and width, respectively, and the maximum value of k is M x N. First, the binary function f of each cover pixel pair ($p_i$, $p_{i+1}$) is defined by f($p_i$,$p_{i+1}$) =LSB(|$p_i$/2|+$p_{i+1}$), where |.| is the floor function and i=1,3,……….,MxN-1. Then, this method embeds each pair of secret bits ($s_i$, $s_{i+1}$) into a cover pixel pair ($p_i$, $p_{i+1}$).

Zhang and Wang [4] proposed an information hiding method named Exploiting Modification Direction(EMD) to embed each secret digit in the base-(2n+1) numeral system into a group of n cover pixels, where n is a positive integer number, by at most one cover pixel is increased or decreased by 1. The EMD method embeds a secret bit stream into a grayscale cover image in spatial domain as follows. Firstly, the secret bit stream is partitioned into segments of l secret bits. Secondly, each secret bit segment is converted into m secret digits in the base-(2n+1) numeral system, where n is a positive integer number and l=|m x $\log_2$(2n+1)|. Next , each time, a group of n consecutive cover pixels($g_1$,$g_2$,………,$g_n$) is taken to embed one secret digit d by using the extraction function h which is defined by h(g1, g2, ..., gn) = ($\sum_{i=1}^{n}$ $g_i$ ×i) mod (2n+1), where mod is the modulo operation. The extraction function values of possible grayscale cover pixel pairs for n = 2 are represented by a unit hyper-cube. It can be any five neighboring elements {h($g_1$, $g_2$), h($g_1$ − 1, $g_2$), h($g_1$ + 1, $g_2$), h($g_1$, $g_2$ −1), h($g_1$, $g_2$ + 1)} are mutually different. This interesting property is the key clue used for the EMD method.

In late 2008, Yang et al. [6] proposed an adaptive least significant bit replacement (A-LSB-R) steganographic method using the pixel-value differencing (PVD) and the minimum-error replacement (MER) technique. Yang et al.'s scheme provides various hiding capacities and good visual qualities of stego images. Yang et al.'s method obeys the basic concept that the edge areas can tolerate more changes than smooth areas. For every two-pixel block ($x_i$, $x_{i+1}$), the difference value between $x_i$ and $x_{i+1}$ is computed by $dv_i$ = |$x_i$ – $x_{i+1}$|. The A-LSB-R method embeds *k* secret bits into each pixel of a cover pixel pair ($x_i$, $x_{i+1}$) at a time. The two-pixel blocks located in the edge areas (i.e., determined by $dv_i$ = |$x_i$ – $x_{i+1}$|) are embedded by a *k*-bit LSB-R method with a larger value of *k* than that of the two-pixel blocks located in smooth areas.

Inspired from Zhang and Wang's scheme, recently, Chang et al.[3] proposed data hiding in digital images using Sudoku puzzle to enhance the hiding capacity. Sudoku is a number placement puzzle. Figure 2 shows the example for Sudoku solution. According to Sudoku properties, Chang et al.'s scheme converts the information to hiding into binary message and again converts into base-9 numeral system digits and then modifies the values of cover pixel pairs for embedding the secret digits with little distortion. Thus, every digit value in the Sudoku grid is decreased by 1 so that the Sudoku grid contains digits from 0 to 8. Next, the modified Sudoku solution is used to generate the reference matrix *R*, as shown in (Annexure) Figure 3, which is used for the embedding and extracting processes [3].

Each secret digit s is embedded into a cover pixel pair (pi, pi+1) as follows. Firstly, the cover pixel pair (pi, pi+1) is located onto the reference matrix R at the row pi and the column pi+1. Secondly, three sets of candidate elements are identified by $C_1$ = {R($p_i$, $_{pi+1}$ – 4), R($p_i$, $p_{i+1}$ – 3), R($p_i$, $p_{i+1}$ – 2), R($p_i$, $p_{i+1}$ – 1), R($p_i$, $p_{i+1}$), R($p_i$, $p_{i+1}$ + 1), R($p_i$, $p_{i+1}$ + 2), R($p_i$, $p_{i+1}$ + 3), R($p_i$, $p_{i+1}$ + 4)}, $C_2$ = {R($p_i$ – 4, $p_{i+1}$), R($p_i$ – 3, $p_{i+1}$), R($p_i$ – 2, $p_{i+1}$), R($p_i$ – 1, pi+1), R(pi, $p_{i+1}$), R($p_{i\ +1}$, $p_{i+1}$), R($p_{i\ +\ 2}$, $p_{i+1}$), R($p_{i\ +\ 3}$, $p_{i+1}$), R($p_{i\ +}$

$_{4}$, $p_{i+1}$)}, and $C_3$ = {R($x_b$, $y_b$), R($x_b$, $y_{b+1}$), R($x_b$, $y_{b+2}$), R($x_{b+1}$, $y_b$), R($x_{b+1}$, $y_{b+1}$), R($x_{b+1}$, $y_{b+2}$), R($x_{b+2}$, $y_b$), R($x_{b+2}$, $y_{b+1}$), R($x_{b+2}$,$y_{b+2}$)}, where xb = $\lfloor pi / 3 \rfloor \times 3$ and $y_b = \lfloor p_{i+1} / 3 \rfloor \times 3$. Next, search from $C_1$, $C_2$, and $C_3$ to find out three candidate elements R($x_h$, $y_h$), R($x_v$, $y_v$), and R($x_w$, $v_w$),respectively, such that R($x_h$, $y_h$) = R($x_v$, $y_v$) = R($x_w$, $v_w$) = s. Then, among the found candidate elements R($x_h$, $y_h$), R($x_v$, $y_v$), and R($x_w$, $v_w$), select the candidate element R($x_f$, $y_f$) with a minimum distortion. That is,R($x_f$, $y_f$) = $\min_j$€{h, v, w}{$|p_i - x_j| + |p_{i+1} - y_j|$}. Finally, the stego pixel pair is obtained by ($p_i$`,$p_{i+1}$`) = ($x_f$, $y_f$). Receiver can extracted the secret message digits from digital media using the received stego-image in reverse procedure [4].

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 5 | 3 | 4 | 6 | 7 | 8 | 9 | 1 | 2 |
| 6 | 7 | 2 | 1 | 9 | 5 | 3 | 4 | 8 |
| 1 | 9 | 8 | 3 | 4 | 2 | 5 | 6 | 7 |
| 8 | 5 | 9 | 7 | 6 | 1 | 4 | 2 | 3 |
| 4 | 2 | 6 | 8 | 5 | 3 | 7 | 9 | 1 |
| 7 | 1 | 3 | 9 | 2 | 4 | 8 | 5 | 6 |
| 9 | 6 | 1 | 5 | 3 | 7 | 2 | 8 | 4 |
| 2 | 8 | 7 | 4 | 1 | 9 | 6 | 3 | 5 |
| 3 | 4 | 5 | 2 | 8 | 6 | 1 | 7 | 9 |

*Figure 2: An Example of a Sudoku Solution*

**Disadvantages**

1. Methodology is very simple
2. Embedding in Red component of the pixel
3. Hiding capacity is very less
4. More computation power is required
5. It is easy for the brute-force
6. Less number of permutations for the key

## 3. PROPOSED SYSTEM

In our proposed scheme, steganography using hexadecimal Sudoku puzzle as mapping soluation to hide secret message into cover image. Any RGB image is used as cover image to hide secret information digits. The secret message can be any media such as text, image, video, audio etc. The digital media is changed from original system to base-16 for mapping into Hexadecimal Sudoku solution.

The proposed scheme implemented in five phases.

I. Generating Hexadecimal Sudoku puzzle.
II. Creating tile matrix from obtained Sudoku
III. Creating 256x256 reference matrix for tile matrix
IV. Mapping the secret message digits with creating reference matrix and digital JPEG image
V. Embedding the secret message digits in image using mapping results

**Advantages**

1. Achieved high embedding capacity.
2. Very less image distortion.
3. It creates the large key space because it uses hexadecimal Sudoku puzzle.
4. Steganalysis is very difficult

### 3.1 Generating a Sudoku

In common 9×9 Sudoku was used in the previous works where it will be easy for a human or a computer to solve the puzzle easily in the minimum available span of time [7][8]. But here 16×16 Sudoku was used where each puzzle may probably have 20922789888000 numbers of solutions for each row. Figure 4 shows the Hexadecimal sudoku solution. This is not only difficult to solve by the humans but also computers may take a lot of time to solve the puzzle, in this way the security was provided.

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | 8 | 4 | 16 | 6 | 10 | 13 | 6 | 3 | 11 | 12 | 15 | 2 | 9 | 14 | 1 |
| 14 | 2 | 15 | 9 | 4 | 12 | 11 | 3 | 16 | 1 | 5 | 13 | 6 | 10 | 8 | 7 |
| 12 | 7 | 6 | 1 | 8 | 15 | 5 | 9 | 4 | 2 | 14 | 10 | 3 | 13 | 11 | 16 |
| 10 | 11 | 13 | 3 | 14 | 1 | 2 | 16 | 6 | 9 | 8 | 7 | 5 | 15 | 4 | 12 |
| 15 | 1 | 3 | 10 | 16 | 2 | 7 | 14 | 13 | 4 | 6 | 12 | 8 | 5 | 9 | 11 |
| 4 | 13 | 16 | 2 | 12 | 5 | 1 | 10 | 9 | 8 | 15 | 11 | 14 | 7 | 6 | 3 |
| 9 | 5 | 7 | 11 | 3 | 8 | 15 | 6 | 2 | 14 | 16 | 1 | 12 | 4 | 13 | 10 |
| 6 | 14 | 12 | 8 | 9 | 13 | 4 | 11 | 5 | 7 | 10 | 3 | 15 | 1 | 16 | 2 |
| 3 | 10 | 8 | 7 | 5 | 14 | 9 | 1 | 15 | 12 | 13 | 6 | 11 | 16 | 1 | 4 |
| 11 | 4 | 14 | 12 | 7 | 6 | 3 | 15 | 1 | 16 | 9 | 2 | 10 | 8 | 5 | 13 |
| 2 | 15 | 9 | 6 | 10 | 16 | 12 | 13 | 11 | 5 | 4 | 8 | 1 | 3 | 7 | 14 |
| 13 | 16 | 1 | 5 | 2 | 11 | 8 | 4 | 10 | 3 | 7 | 14 | 9 | 6 | 12 | 15 |
| 7 | 6 | 10 | 13 | 1 | 9 | 14 | 12 | 8 | 15 | 11 | 16 | 4 | 2 | 3 | 5 |
| 8 | 9 | 5 | 15 | 11 | 3 | 16 | 2 | 12 | 13 | 1 | 4 | 7 | 14 | 10 | 6 |
| 1 | 3 | 11 | 4 | 13 | 7 | 10 | 8 | 14 | 6 | 2 | 5 | 16 | 12 | 15 | 9 |
| 16 | 12 | 2 | 14 | 15 | 4 | 6 | 5 | 7 | 10 | 3 | 9 | 13 | 11 | 1 | 8 |

*Figure 4: Hexa(16×16) Sudoku*

### 3.2 Creating a Tile Matrix from Obtained Sudoku

Sudoku which was generated in the above will be taken in a Matrix for easy usage in the further work. After generated a Sudoku a Tile Matrix was created by subtracting 1 from each digit in puzzle to make the range from 0-E, for mapping secret digits and Sudoku puzzle. Figure 5 represents the Title matrix, which is used to generate the reference matrix.

### 3.3 Creating a Reference Matrix

In this phase from the created Tile Matrix, 256x256 reference matrix is generated to calculate the modification candidate components in the cover image. Figure 6 shows the reference matrix (M). In this process the Tile matrix is repeated 16 times horizontally and 16 times vertically without doing any medications in tile matrix. The resultant matrix referred as Reference Matrix. It used to embed the secret messages digits in cover image.

### 3.4 Calculating Components

It takes the cover image as input. Pixel by pixel of the cover image is extracted, particularly the red and green values of the pixels. The color values of pixel ranges from 0 to 255. Here R is the value of Red color in the picture, and G is the value of green color in the picture. They are modified as above in order to maintain compatibility between Sudoku values, Secret data and the pixel values. Now the values of R, G are represented as $g_i$, $g_{i+1}$ respectively.

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 7 | 3 | 15 | 5 | 9 | 12 | 6 | 2 | 10 | 11 | 14 | 1 | 8 | 13 | 0 |
| 13 | 1 | 14 | 8 | 3 | 11 | 10 | 2 | 15 | 0 | 4 | 12 | 5 | 9 | 7 | 6 |
| 11 | 6 | 5 | 0 | 7 | 14 | 4 | 8 | 3 | 1 | 13 | 9 | 2 | 12 | 10 | 15 |
| 9 | 10 | 12 | 2 | 13 | 0 | 1 | 15 | 5 | 8 | 7 | 6 | 4 | 14 | 3 | 11 |
| 14 | 0 | 2 | 9 | 15 | 1 | 6 | 13 | 12 | 3 | 5 | 11 | 7 | 4 | 8 | 10 |
| 3 | 12 | 15 | 1 | 11 | 4 | 0 | 9 | 8 | 7 | 14 | 10 | 13 | 6 | 5 | 2 |
| 8 | 4 | 6 | 10 | 2 | 7 | 14 | 5 | 1 | 13 | 15 | 0 | 11 | 3 | 12 | 9 |
| 5 | 13 | 11 | 7 | 8 | 12 | 3 | 10 | 4 | 6 | 9 | 2 | 14 | 0 | 15 | 1 |
| 2 | 9 | 7 | 6 | 4 | 13 | 8 | 0 | 14 | 11 | 12 | 5 | 10 | 15 | 1 | 3 |
| 10 | 3 | 13 | 11 | 6 | 5 | 2 | 14 | 0 | 15 | 8 | 1 | 9 | 7 | 4 | 12 |
| 1 | 14 | 8 | 5 | 9 | 15 | 11 | 12 | 10 | 4 | 3 | 7 | 0 | 2 | 6 | 13 |
| 12 | 15 | 0 | 4 | 1 | 10 | 7 | 3 | 9 | 2 | 6 | 13 | 8 | 5 | 11 | 14 |
| 6 | 5 | 9 | 12 | 0 | 8 | 13 | 11 | 7 | 14 | 10 | 15 | 3 | 1 | 2 | 4 |
| 7 | 8 | 4 | 14 | 10 | 2 | 15 | 1 | 11 | 12 | 0 | 3 | 6 | 13 | 9 | 5 |
| 0 | 2 | 10 | 3 | 12 | 6 | 9 | 7 | 13 | 5 | 1 | 4 | 15 | 11 | 14 | 8 |
| 15 | 11 | 1 | 13 | 14 | 3 | 5 | 4 | 6 | 9 | 2 | 8 | 12 | 10 | 0 | 7 |

*Figure 5 :16×16 Tile Matrix Sudoku(T)*

Then 3 mapping candidate elements are chosen called Horizontal ($CE_H$), Vertical ($CE_V$), and Boxed ($CE_B$). The required Mathematical notations for selecting values from the reference matrix are as follows:

**For Horizontal component ($CE_H$):**

**S1:**

**If $p_{i+1} > 5$and $p_{i+1} < 246$ Then**

$CE_H = \{M(p_i, p_{i+1} - 5), M(pi, p_{i+1} - 4), M(p_i, p_{i+1} - 3), M(p_i, p_{i+1} - 2), M(p_i, p_{i+1}-1), M(p_i, p_{i+1}), M(p_i, p_{i+1} + 1), M(p_i, p_{i+1} +2), M(p_i, p_{i+1} +3), M(p_i, p_{i+1} +4), M(p_i, p_{i+1} +5), M(p_i, p_{i+1} +6), M(p_i, p_{i+1} +7), M(p_i, p_{i+1} +8), M(p_i, p_{i+1} +9), M(p_i, p_{i+1} +10)\};$

**else If $p_{i+1} \leq 5$ Then**

$CE_H = \{M(p_i, p_{i+1}), M(p_i, p_{i+1}+1), M(p_i, p_{i+1}+2), M(p_i, p_{i+1}+3), M(p_i, p_{i+1}+4), M(p_i, p_{i+1}+5), M(p_i, p_{i+1}+6), M(p_i, p_{i+1}+7), M(p_i, p_{i+1}+8), M(p_i, p_{i+1}+9), M(p_i, p_{i+1}+10), M(p_i, p_{i+1}+11), M(p_i, p_{i+1}+12), M(p_i, p_{i+1}+13), M(p_i, p_{i+1}+14), M(p_i, p_{i+1}+15)\}$

**else If $p_{i+1} \geq 246$ Then**

$CE_H = \{M(p_i, p_{i+1}), M(p_i, p_{i+1}-1), M(p_i, p_{i+1}-2), M(p_i, p_{i+1}-3), M(p_i, p_{i+1}-4), M(p_i, p_{i+1}-5), M(p_i, p_{i+1}-6), M(p_i, p_{i+1}-7), M(p_i, p_{i+1}-8), M(p_i, p_{i+1}-9), M(p_i, p_{i+1}-10), M(p_i, p_{i+1}-11), M(p_i, p_{i+1}-12), M(p_i, p_{i+1}-13), M(p_i, p_{i+1}-14), M(p_i, p_{i+1}-15)\}$

**For Vertical Component ($CE_V$):**

**S2:**

**If $p_i > 5$and $p_i < 246$ Then**

$CE_V = \{M(p_i -5, p_{i+1}), M(p_i-4, p_{i+1}), M(p_i-3, p_{i+1}), M(p_i-2, p_{i+1}), M(p_i-1, p_{i+1}), M(p_i, p_{i+1}), M(p_i+1, p_{i+1}), M(p_i +2, p_{i+1}), M(p_i +3, p_{i+1}), M(p_i +4, p_{i+1}), M(p_i +5, p_{i+1}), M(p_i +6, p_{i+1}), M(p_i +7, p_{i+1}), M(p_i +8, p_{i+1}), M(p_i +9, p_{i+1}), M(p_i +10, p_{i+1})\}$

**else If $p_i \leq 5$ then**

$CE_V = \{M(p_i,p_{i+1}), M(p_i+1,p_{i+1}), M(p_i+2,p_{i+1}), M(p_i+3,p_{i+1}), M(p_i+4,p_{i+1}), M(p_i+5,p_{i+1}), M(p_i+6,p_{i+1}), M(p_i+7,p_{i+1}), M(p_i+8,p_{i+1}), M(p_i+9,p_{i+1}), M(p_i+10,p_{i+1}), M(p_i+11,p_{i+1}), M(p_i+12,p_{i+1}), M(p_i+13,p_{i+1}), M(p_i+14,p_{i+1}), M(p_i+15,p_{i+1}) \};$

**else If $p_i \geq 246$ then**

$CE_V = \{M(p_i-1, p_{i+1}), M(p_i-2, p_{i+1}), M(p_i-3, p_{i+1}), M(p_i-4, p_{i+1}), M(p_i-5, p_{i+1}), M(p_i-6, p_{i+1}), M(p_i-7, p_{i+1}), M(p_i-8, p_{i+1}), M(p_i-9, p_{i+1}), M(p_i-10, p_{i+1}), M(p_i-11, p_{i+1}), M(p_i-12, p_{i+1}), M(p_i-13, p_{i+1}), M(p_i-14, p_{i+1}), M(p_i-15, p_{i+1}), M(p_i-16, p_{i+1}) \}.$

**For Boxed Component ($CE_B$):**

**S3:**

**If $p_i<251$ and $p_{i+1} <251$ then**

**$x_b= \lfloor p_i/4 \times 4 \rfloor$ , $y_b= \lfloor p_{i+1} / 4 \rfloor \times 4$**

$CE_B = \{M(x_{b,}y_b), M(x_{b,}y_{b+1}), M(x_{b,}y_{b+2}), M(x_{b,}y_{b+3}), M(x_{b+1,}y_b), M(x_{b+1,}y_{b+1}), M(x_{b+1,}y_{b+2}), M(x_{b+1,}y_{b+3}), M(x_{b+2,}y_b), M(x_{b+2,}y_{b+1}), M(x_{b+2,}y_{b+2}), M(x_{b+2,}y_{b+3})\}$

**else**

$CE_B$ is empty

### 3.5 Embedding and De-embedding Data

#### Embedding

The secret digit to be embedded location is identified in the cover image with respect to Red ($p_i$) and Green($p_{i+1}$) values of pixel. With respect to secret digit $S_i$, three candidate elements $M(x_H, y_H)$, $M(x_V, y_V)$, and $M(x_B, y_B)$ are calculated form $CE_H$, $CE_V$, and $CE_B$, respectively. The Red and Green values are less than 255, then $S_i = M(x_H, y_H) = M(x_V, y_V) = M(x_B, y_B)$; otherwise, $S_{i=}M(x_H, y_H) = M(x_V, y_V)$. The selected pixel in the cover image $(g_i, g_{i+1})$ is modified $(p`_i, p`_{i+1})$ by a minimum candidate element $M(x_{min}, y_{min})$ which is selected by using

Manhattan distance formula: $M(x_{min}, y_{min}) = \min_j = H,V,B\{| p_i - x_j | + |p_{i+1} - y_j|\}$.

Thus, the cover image pixel pair $(p_i, p_{i+1})$ is modified as $(p`_i = x_{min}, p`_{i+1} = y_{min})$ to hide the secret digit $S_i$ with little distortion. This little distortion does not affect the cover image much and physically no difference or change can be identified of this digital image with respect to cover image. The same procedure applied to all the pixels of the cover image until all digits of secret message embedded.

For more complexity is added to the secret digits using the encryption. Before embedding phase, secret message is encrypted using any one encryption technique. In the proposal scheme, the main requirement is to share the common Sudoku solution to both sender and receiver. In addition, the receiver should aware the media used for message embedding. Figure 7 emphases the embedding process.

**Data De-Embedding**

In this phase, the embedded secret message digits are extracted from the cover image. Figure 8 explains the de-embedding process. It is reverse process of the embedding phase. The Sudoku solution which is shared between the sender and the receiver is shown in (Annexure) Figure 6. This solution forming pair $(p_i, p_{i+1})$, where $p_i$ = R and $p_{i+1}$ = G. The value at position $(p_i, p_{i+1})$ is the secret message digit. This process is applied for all pixels and until secret message retrieved. The obtained secret digits are converted from base-16 to base-9 number system. This completes De-Embedding phase.

**Algorithm of the process**

**Embedding Algorithm**

**Step 1:** Generate a 16×16 Sudoku.

**Step 2** : Generate Tile Matrix Sudoku, by subtracting 1 from all numbers

**Step 3**: Generate a 256×256 Reference Matrix Sudoku.

**Step 4**: Read necessary Pixels from the image.

**Step 5**: Select Horizontal, Vertical, Boxed Components.

**Step 6**: Read the input string.

**Step 7**: Convert input String to hexadecimal code.

**Step 8**: Select the required elements in reference matrix in all components.

**Step 9**: Calculate the Manhattan distance with the least value.

**Step 10**: Replace the pixel values with the respective values obtained.

**Output**: Stego-image.

In embedding phase, the secret message stream converted into hexadecimal number system, which represents one character in secret message in two hexadecimal digits. The converted secret hexadecimal stream is denoted as $S=s_1,s_2,\ldots\ldots,s_n$, where n is the number of hexadecimal digits in the secret digit stream. For example $(p_i,p_{i+1})$ is $p_i$ pixel's red and green pixel pair values for mapping secret digit $s_i$. In this process, first select $m_i=M[p_i,p_{i+1}]$ from the Reference matrix(R), which is generated form Sudoku solution. Once the location $m_i$ has identified in R, three sets of candidate elements $CE_H$, $CE_V$, $CE_B$ can be constructed accordingly to the formulas given above. The values in the three sets consists {1,2,3,……..15}. Based on the value $s_i$, three positions are extracted from the three candidate elements. The positions are denoted as $M(X_H,Y_H)$, $M(X_V,Y_V)$, $M(X_B,Y_B)$.

**Embedding Flow chart**

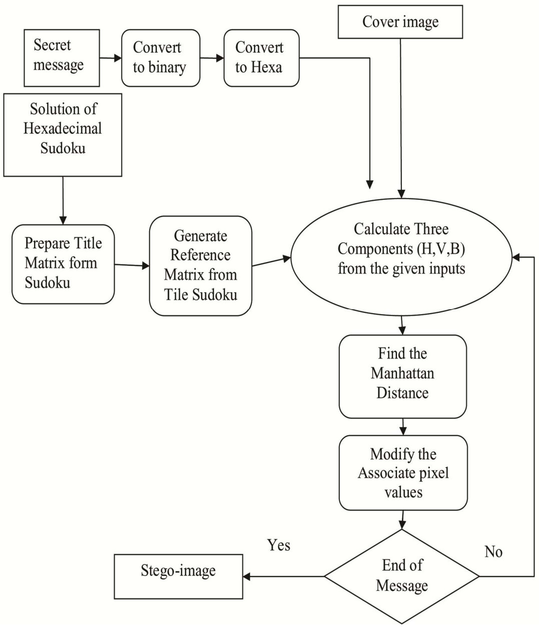

Secret message
Convert to binary
Convert to Hexa
Cover image
Solution of Hexadecimal Sudoku
Prepare Title Matrix form Sudoku
Generate Reference Matrix from Tile Sudoku
Calculate Three Components (H,V,B) from the given inputs
Find the Manhattan Distance
Modify the Associate pixel values
Yes
No
Stego-image
End of Message

*Figure 7: The Diagram Of The Proposed Embedding Process.*

The embedding is done by modifying pixel pair $(p_i,p_{i+1})$ to $(p`_i, p`_{i+1})$ based on the minimum Manhattan distance between $(p_{i1},p_{i2})$ and these three candidates, i.e.,

$M(x_{min}, y_{min}) = \min_j = H,V,B\{| p_i - x_j | + |p_{i+1} - y_j|\}$

Once the candidate pair min min $(x \ ,y \ )$with minimum Manhattan distance has been obtained, the cover pixel pair $(p_i,p_{i+1})$ is then modified to $(x_{min}, y_{min})$ for concealing a secret digit $s_i$, i.e., $(p`_i, p`_{i+1}) \leftarrow (x_{min}, y_{min})$. Here is a simple example to illustrate the data embedding procedures.

**Extracting Algorithm**

**Step 1**: Generate a 16×16 SUDOKU.
**Step 2**: Generate TILE Matrix Sudoku, by subtracting 1 from all numbers.
**Step 3**: Generate a 256×256 Reference Matrix Sudoku.
**Step 4**: Read necessary Pixel from the stego-image.
**Step 5**: Extract the Pixel's Red and Green Values.
**Step 6**: Consider the Red as row and Green as column indexes of Reference Matrix.
**Step 7**: Store the hexadecimal value, which is extracted from the Reference Matrix.
**Step 8**: Convert the hexadecimal value into Binary.
**Step 9**: Convert the binary into ASCII Characters.
**Output**: Secret Message

**Extracting Flow chart**

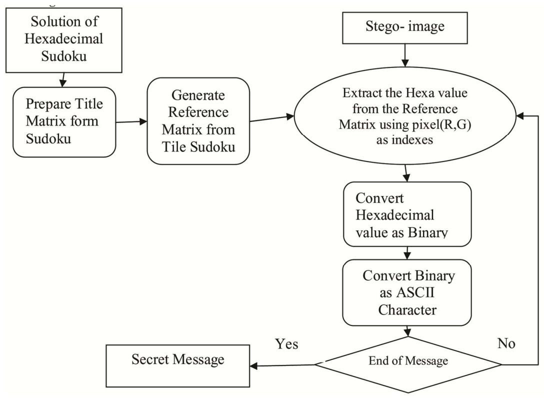

Solution of Hexadecimal Sudoku
Prepare Title Matrix form Sudoku
Generate Reference Matrix from Tile Sudoku
Stego- image
Extract the Hexa value from the Reference Matrix using pixel(R,G) as indexes
Convert Hexadecimal value as Binary
Convert Binary as ASCII Character
End of Message
Yes
No
Secret Message

*Figure 8: The Diagram Of The Proposed De-Embedding Process*

## 4. SIMULATION RESULTS

Simulation was performed in Matlab & the experimental results are presented in this section. In high bit-rate data hiding, we have two primary objectives, viz, the method should provide the maximum possible payload and the embedded data must be unknown to the observer [11]. We stress on the fact that steganography is not meant to be robust. Any modifications to the file, such as conversions between file types and/or standard image processing, is expected to remove the hidden bits from the file. Fundamentally, data payload of a steganographic scheme can be defined as the amount of secret message can hide within the cover image. As with any method of storing data, this can be expressed as a number of bits, which indicates the max message size that might be inserted into an image. If we assumed that the color image contains XY pixels then every sub band of its wavelet transform will contain 3*(XY/4) coefficients. So the data payload of the algorithm presented here can be expressed using equations (1) & (2) as

$$\text{Data Payload}= 3\,x\,4\left(XY\ \ /\,4\right)xN \quad \text{bits} \quad (1)$$

Payload % age=

$$\frac{3x4(XY/4)x\frac{N}{8}}{3XY}x100\% = \left(\frac{N}{8}x100\right)\% \quad (2)$$

Obviously, the presented scheme is blind since with the stego key only, the original cover-image is not needed to recover the embedded message from the received stego-image. In addition, the proposed scheme is considered secure. That is, without knowing the stego-key a passive warden can't extract the hidden message or even prove its very existence. Usually the invisibility of the hidden message is measured in terms of the Peak Signal-to-Noise Ratio (PSNR) defined in equation (3), where p(x,y) represents the color shade level of a pixel, whose coordinates are (x , y) in the selected image, and ~p(x, y) represents the same pixel in the distorted image. RMSE represents the Root Mean Square Error (RMSE) as a measurement criterion [9].

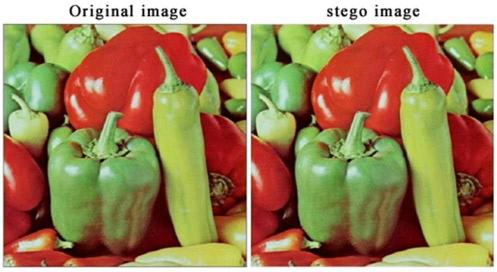
$$\text{PSNR}=10\ \log_{10}\left(\frac{L^2}{MSE}\right)_{db} \quad (3)$$

$$\text{MSE}=\frac{1}{XY}\sum_{X,Y}\left(p(s,y)-\tilde{p}(x,y)\right)^2 \quad (4)$$

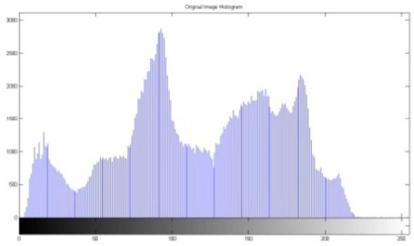
$$\text{RMSE}=\sqrt{\frac{1}{XY}\sum_{X,Y}\left(p(s,y)-\tilde{p}(x,y)\right)^2} \quad (5)$$

Two sets of experimental results are presented here

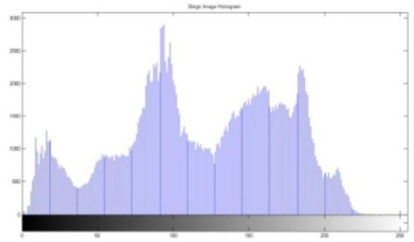
**Experimental results 1:**



*Peppers Original Image* *Peppers Stego Image*



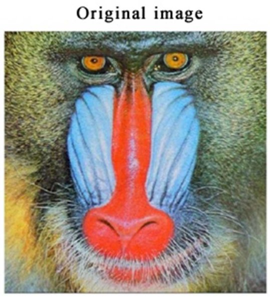
*Peppers Original image Histogram*



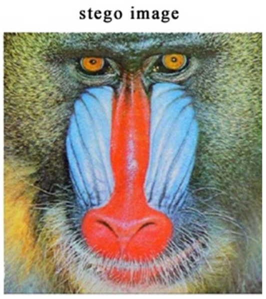
*Peppers Stego Image Histogram*

Image size: 512X512
Secret Data size: 28 KB
PSNR=45.65
MSE=5.31
RMSE=1.7639
INFERENCE: The PSNR value is between the optimal value (35db-45db) and the mean square error is minimum.

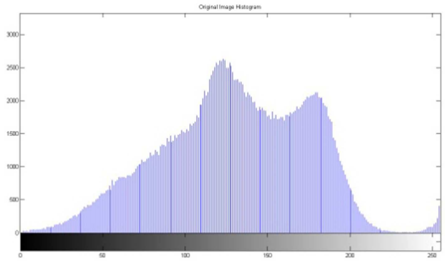
**Experimental Results 2:**



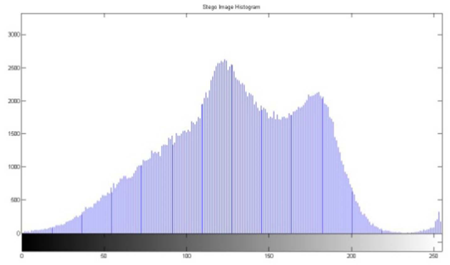
*Baboon Original Image*



*Baboon Stego Image*



*Baboon Original Image Histogram*

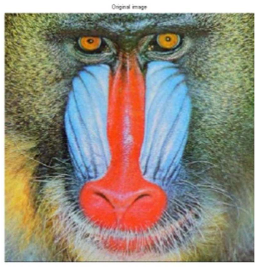


*Baboon Stego Image Histogram*

Image size:512x512
Secret Data size:8KB
MSE:1.42
PSNR:51.18
RMSE:1.1624
INFERENCE:The PSNR value is between the optimal value(35db-45db) and the mean square error is minimum.

**Experiment Result 3:**

The proposed method applied on four types of images (jpg, bmp, tif, png) sized 512x512 to evaluate the performance in terms of visual quality (distortion rate) of stego image and hiding capacity. The visual quality of stego image was evaluated by using peak signal-to-noise ratio (PSNR) which is defined as PSNR = 10×log10(2552 / MSE) (dB), where MSE is the mean square error representing the distortion between the cover image (X) sized HxW and the stego image(V) sized H xW. Hiding capacity(C) was measured by bits per pixel (bpp). That is, it was computed by the ratio of the total number of embedded secret bits and the total number of pixels in the cover image. The hiding capacities and the PSNR values of the proposed method with various values of C for test images are shown in Table 1.



*Baboon Image (512X512)*

It is very difficult to the steganalysis to discover the hidden message from stego image. if the cover image is modified more significantly in order to embed the secret message, that may give the change to detectors detected the hidden message. Figure 9 shows the analyzed results on hiding capacity and visual quality of the image.

*Table 1: Hiding capacities and PSNR values of the proposed method for test images with various values of capacity(C).*

| Type / Image | JPG | | TIF | | BMP | | PNG | |
|---|---|---|---|---|---|---|---|---|
| | PSNR | C (KB) | PSNR | C (KB) | PSNR | C (KB) | PSNR | C (KB) |
| **Baboon (512X512)** | 41.20 | 1.36 | 59.25 | 1.36 | 59.07 | 1.36 | 58.62 | 1.36 |
| | 40.82 | 7.18 | 51.66 | 7.18 | 51.63 | 7.18 | 51.30 | 7.18 |
| | 40.63 | 10.20 | 50.14 | 10.20 | 50.08 | 10.20 | 49.87 | 10.20 |
| | 40.57 | 12.70 | 49.15 | 12.70 | 49.11 | 12.70 | 48.95 | 12.70 |
| | 40.53 | 14.70 | 48.57 | 14.70 | 48.52 | 14.70 | 48.37 | 14.70 |
| | 40.53 | 15.70 | 48.23 | 15.70 | 48.20 | 15.70 | 48.06 | 15.70 |
| | 40.44 | 19.10 | 47.23 | 19.10 | 47.28 | 19.10 | 47.28 | 19.10 |
| | 40.40 | 20.00 | 47.18 | 20.00 | 47.16 | 20.00 | 47.04 | 20.00 |
| | 40.19 | 27.10 | 45.89 | 27.10 | 45.82 | 27.10 | 45.75 | 27.10 |

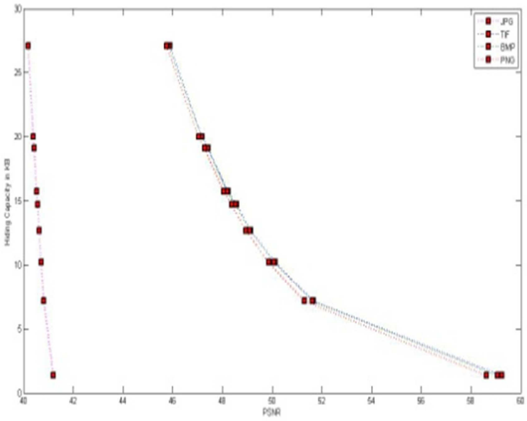


*Figure 9: Comparison results on cover image based on degree of modification.*

From the results of plot analyses, we can see the distortion rate (visual quality) of image and hiding capacity. Thus, the proposed method does not create a significant image distortion on the cover image.

## 5. CONSLUSION

The proposed method generated better results compared to existing methods based on the degree of modification on cover image. This paper is addressed improved results on embedding capacity to achieve the little distortion and prevents the steganlysis to identify the secret message digits. Any steganographic technique should improve the embedding capacity without losing the visual quality of the image. In fact visual quality of the digital image is inversely proportional to the hiding capacity of the secret message. The proposed method improved the hiding capacity with nominal distortion. This little distortion will not give the chance to the steganalysis to discover the hidden message. It prevents all the discovered methods even degree of hiding capacity improved. Because this method is not modifying the part of pixel, it is modifying the entire pixel value based on the mapped values identified with the Sudoku solution. In this scheme, we are modified only red and green values of pixel. In future, we will plan to modify the blue color value of pixel also, that may improve the more hiding capacity.

## References:

[1] Amitava Nag et.al. “A Huffman Code Based Image Steganography Technique”. ICAA 2014: 257-265.

[2] A.Cheddad, J.Condell, K.Curran and P.Mc Kevitt, “Digital Image Steganography: Survey and Analysis of current methods”. Signal Processing, Volume 90, Issue 3, March 2010, Pages 727-752.

[3] C.C. Chang, T. D. Kieu, and Y.-C. Chou. “High capacity data hiding for grayscale images”. In *Proceedings of the First International Conference on Ubiquitous Information Management and Communication*, pages 139–148. Seoul, Korea, February 2007.

[4] Roshan Shetty B R, Rohith J, Mukund V, Rohan Honwade, Shanta Rangaswamy. Steganography Using Sudoku Puzzle. 2009 International Conference on Advances in Recent Technologies in Communication and Computing, pages 623-626.

[5] Y.T. Wu and F. Y. Shih. Digital watermarking based on chaotic map and reference register. Pattern Recognition, 40(12):3754–3763, December 2007.

[6] C.C. Chang, Y.C. Chou and T.D. Kieu, An Information Hiding Scheme Using Sudoku, Proceedings of the Third International Conference on Innovative Computing, Information and Control (ICICIC2008), June 2008.

[7] B. Felgenhauer and F. Jarvis. Mathematics of Sudoku I, Mathematical Spectrum, vol. 39, no. 1, pp. 15-22, 2006. 626.

[8] Felgenhauer and F. Jarvis. Mathematics of Sudoku I, Mathematical Spectrum, vol. 39, no. 1, pp. 15-22, 2006. 626.

[9] Wikipedia, URL: http://www.wikipedia.org.

[10] Moerland T., “Steganography and Steganalysis”, *Leiden Institute of Advanced Computing Science*, www.liacs.nl/home/tmoerl/privtech.pdf.

[11] Silman, J., “Steganography and Steganalysis: An Overview”, *SANS Institute*, 2001

[12] Jamil, T., “Steganography: The art of hiding information is plain sight”, *IEEE Potentials*, 18:01, 1999.

[13] Wang, H & Wang, S, “Cyber warfare: Steganography vs. Steganalysis”, *Communications of the ACM*, 47:10, October 2004.

[14] Anderson, R.J. & Petitcolas, F.A.P., “On the limits of steganography”, *IEEE Journal of selected Areas in Communications*, May 1998.

[15] Marvel, L.M., Boncelet Jr., C.G. & Retter. C., “Spread Spectrum Steganography”, *IEEE Transactions on image processing*, *8:08*, 1999.

[16] Artz, D., “Digital Steganography: Hiding Data within Data”, *IEEE Internet Computing Journal*, June 2001.

[17] Currie, D.L. & Irvine, C.E., “Surmounting the effects of lossy compression on Steganography”, *19th National Information Systems Security Conference*, 1996.

[18] Johnson, N.F. & Jajodia, S., “Exploring Steganography: Seeing the Unseen”, *Computer Journal*,February 1998.

[19] Petitcolas, F.A.P., Anderson, R.J. & Kuhn, M.G., “Information Hiding – A survey”, *Proceedings of the IEEE*, 87:07, July 1999.

[20] Arup Kumar Pal, Tarok Pramanik., “Design of an Edge Detection Based Image Steganography with High Embedding Capacity”, QSHINE 2013: 794-800.

## ANNEXURE

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | -- | 252 | 253 | 224 | 225 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 4 | 2 | 3 | 5 | 6 | 7 | 8 | 0 | 1 | 4 | 2 | 3 | 5 | 6 | 7 | 8 | 0 | 1 | -- | 4 | 2 | 3 | 5 |
| 1 | 5 | 6 | 1 | 0 | 8 | 4 | 2 | 3 | 7 | 5 | 6 | 1 | 0 | 8 | 4 | 2 | 3 | 7 | -- | 5 | 6 | 1 | 0 |
| 2 | 0 | 8 | 7 | 2 | 3 | 1 | 4 | 5 | 6 | 0 | 8 | 7 | 2 | 3 | 1 | 4 | 5 | 6 | -- | 0 | 8 | 7 | 2 |
| 3 | 7 | 4 | 8 | 6 | 5 | 0 | 3 | 1 | 2 | 7 | 4 | 8 | 6 | 5 | 0 | 3 | 1 | 2 | -- | 7 | 4 | 8 | 6 |
| 4 | 3 | 1 | 5 | 7 | 4 | 2 | 6 | 8 | 0 | 3 | 1 | 5 | 7 | 4 | 2 | 6 | 8 | 0 | -- | 3 | 1 | 5 | 7 |
| 5 | 6 | 0 | 2 | 8 | 1 | 3 | 7 | 4 | 5 | 6 | 0 | 2 | 8 | 1 | 3 | 7 | 4 | 5 | -- | 6 | 0 | 2 | 8 |
| 6 | 8 | 5 | 0 | 4 | 2 | 6 | 1 | 7 | 3 | 8 | 5 | 0 | 4 | 2 | 6 | 1 | 7 | 3 | -- | 8 | 5 | 0 | 4 |
| 7 | 1 | 7 | 6 | 3 | 0 | 8 | 5 | 2 | 4 | 1 | 7 | 6 | 3 | 0 | 8 | 5 | 2 | 4 | -- | 1 | 7 | 6 | 3 |
| 8 | 2 | 3 | 4 | 1 | 7 | 5 | 0 | 6 | 8 | 2 | 3 | 4 | 1 | 7 | 5 | 0 | 6 | 8 | -- | 2 | 3 | 4 | 1 |
| 9 | 4 | 2 | 3 | 5 | 4 | 2 | 3 | 5 | 4 | 2 | 3 | 5 | 4 | 2 | 3 | 5 | 4 | 2 | -- | 4 | 2 | 3 | 5 |
| 10 | 5 | 6 | 1 | 0 | 5 | 6 | 1 | 0 | 5 | 6 | 1 | 0 | 5 | 6 | 1 | 0 | 5 | 6 | -- | 5 | 6 | 1 | 0 |
| 11 | 0 | 8 | 7 | 2 | 0 | 8 | 7 | 2 | 0 | 8 | 7 | 2 | 0 | 8 | 7 | 2 | 0 | 8 | -- | 0 | 8 | 7 | 2 |
| 12 | 7 | 4 | 8 | 6 | 7 | 4 | 8 | 6 | 7 | 4 | 8 | 6 | 7 | 4 | 8 | 6 | 7 | 4 | -- | 7 | 4 | 8 | 6 |
| 13 | 3 | 1 | 5 | 7 | 4 | 2 | 6 | 8 | 0 | 3 | 1 | 5 | 7 | 4 | 2 | 6 | 8 | 0 | -- | 3 | 1 | 5 | 7 |
| 14 | 6 | 0 | 2 | 8 | 1 | 3 | 7 | 4 | 5 | 6 | 0 | 2 | 8 | 1 | 3 | 7 | 4 | 5 | -- | 6 | 0 | 2 | 8 |
| 15 | 8 | 5 | 0 | 4 | 2 | 6 | 1 | 7 | 3 | 8 | 5 | 0 | 4 | 2 | 6 | 1 | 7 | 3 | -- | 8 | 5 | 0 | 4 |
| 16 | 1 | 7 | 6 | 3 | 0 | 8 | 5 | 2 | 4 | 1 | 7 | 6 | 3 | 0 | 8 | 5 | 2 | 4 | -- | 1 | 7 | 6 | 3 |
| 17 | 2 | 3 | 4 | 1 | 7 | 5 | 0 | 6 | 8 | 2 | 3 | 4 | 1 | 7 | 5 | 0 | 6 | 8 | -- | 2 | 3 | 4 | 1 |
| : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : |
| 252 | 4 | 2 | 3 | 5 | 4 | 2 | 3 | 5 | 4 | 2 | 3 | 5 | 4 | 2 | 3 | 5 | 4 | 2 | -- | 4 | 2 | 3 | 5 |
| 243 | 5 | 6 | 1 | 0 | 5 | 6 | 1 | 0 | 5 | 6 | 1 | 0 | 5 | 6 | 1 | 0 | 5 | 6 | -- | 5 | 6 | 1 | 0 |
| 254 | 0 | 8 | 7 | 2 | 0 | 8 | 7 | 2 | 0 | 8 | 7 | 2 | 0 | 8 | 7 | 2 | 0 | 8 | -- | 0 | 8 | 7 | 2 |
| 255 | 7 | 4 | 8 | 6 | 7 | 4 | 8 | 6 | 7 | 4 | 8 | 6 | 7 | 4 | 8 | 6 | 7 | 4 | -- | 7 | 4 | 8 | 6 |

*Figure 3: An example of the Reference Matrix R*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | -- | 253 | 254 | 255 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 4 | 7 | 3 | 15 | 5 | 9 | 12 | 6 | 2 | 10 | 11 | 14 | 1 | 8 | 13 | 0 | -- | 8 | 13 | 0 |
| 1 | 13 | 1 | 14 | 8 | 3 | 11 | 10 | 2 | 15 | 0 | 4 | 12 | 5 | 9 | 7 | 6 | -- | 9 | 7 | 6 |
| 2 | 11 | 6 | 5 | 0 | 7 | 14 | 4 | 8 | 3 | 1 | 13 | 9 | 2 | 12 | 10 | 15 | -- | 12 | 10 | 15 |
| 3 | 9 | 10 | 12 | 2 | 13 | 0 | 1 | 15 | 5 | 8 | 7 | 6 | 4 | 14 | 3 | 11 | -- | 14 | 3 | 11 |
| 4 | 14 | 0 | 2 | 9 | 15 | 1 | 6 | 13 | 12 | 3 | 5 | 11 | 7 | 4 | 8 | 10 | -- | 4 | 8 | 10 |
| 5 | 3 | 12 | 15 | 1 | 11 | 4 | 0 | 9 | 8 | 7 | 14 | 10 | 13 | 6 | 5 | 2 | -- | 6 | 5 | 2 |
| 6 | 8 | 4 | 6 | 10 | 2 | 7 | 14 | 5 | 1 | 13 | 15 | 0 | 11 | 3 | 12 | 9 | -- | 3 | 12 | 9 |
| 7 | 5 | 13 | 11 | 7 | 8 | 12 | 3 | 10 | 4 | 6 | 9 | 2 | 14 | 0 | 15 | 1 | -- | 0 | 15 | 1 |
| 8 | 2 | 9 | 7 | 6 | 4 | 13 | 8 | 0 | 14 | 11 | 12 | 5 | 10 | 15 | 1 | 3 | -- | 15 | 1 | 3 |
| 9 | 10 | 3 | 13 | 11 | 6 | 5 | 2 | 14 | 0 | 15 | 8 | 1 | 9 | 7 | 4 | 12 | -- | 7 | 4 | 12 |
| 10 | 1 | 14 | 8 | 5 | 9 | 15 | 11 | 12 | 10 | 4 | 3 | 7 | 0 | 2 | 6 | 13 | -- | 2 | 6 | 13 |
| 11 | 12 | 15 | 0 | 4 | 1 | 10 | 7 | 3 | 9 | 2 | 6 | 13 | 8 | 5 | 11 | 14 | -- | 5 | 11 | 14 |
| 12 | 6 | 5 | 9 | 12 | 0 | 8 | 13 | 11 | 7 | 14 | 10 | 15 | 3 | 1 | 2 | 4 | -- | 1 | 2 | 4 |
| 13 | 7 | 8 | 4 | 14 | 10 | 2 | 15 | 1 | 11 | 12 | 0 | 3 | 6 | 13 | 9 | 5 | -- | 13 | 9 | 5 |
| 14 | 0 | 2 | 10 | 3 | 12 | 6 | 9 | 7 | 13 | 5 | 1 | 4 | 15 | 11 | 14 | 8 | -- | 11 | 14 | 8 |
| 15 | 15 | 11 | 1 | 13 | 14 | 3 | 5 | 4 | 6 | 9 | 2 | 8 | 12 | 10 | 0 | 7 | -- | 10 | 0 | 7 |
| : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : | : |
| 253 | 7 | 8 | 4 | 14 | 10 | 2 | 15 | 1 | 11 | 12 | 0 | 3 | 6 | 13 | 9 | 5 | -- | 13 | 9 | 5 |
| 254 | 0 | 2 | 10 | 3 | 12 | 6 | 9 | 7 | 13 | 5 | 1 | 4 | 15 | 11 | 14 | 8 | -- | 11 | 14 | 8 |
| 255 | 15 | 11 | 1 | 13 | 14 | 3 | 5 | 4 | 6 | 9 | 2 | 8 | 12 | 10 | 0 | 7 | -- | 10 | 0 | 7 |

*Figure 6: 256×256 Reference Matrix Sudoku(M)*